OPT++

A Toolkit for Nonlinear Optimization

Juan Meza

High Performance Computing Research

Lawrence Berkeley National Laboratory

http://www.nersc.gov/~meza

Acknowledgements

- OPT++ is an open source toolkit for general nonlinear optimization problems
- Original development started in 1992 at SNL/CA
- Major contributors
  - Juan Meza, LBNL
  - Patty Hough, SNL/CA
  - Pam Williams, SNL/CA
- Other members of the OPT++ team
  - Vicki Howle
  - Kevin Long
  - Suzanne Shontz

Outline

- Introduction to Optimization
- OPT++ Philosophy
- Classes of Optimization Solvers
- Setting up a Problem and Algorithm
- Example 1: Unconstrained Optimization
- Example 2: Constrained Optimization
- Parallel optimization techniques
- Summary

General Optimization Problem

$$\min_{x \in \Re^n} f(x),$$ Objective function

$$s.t. \quad h(x) = 0,$$ Equality constraints

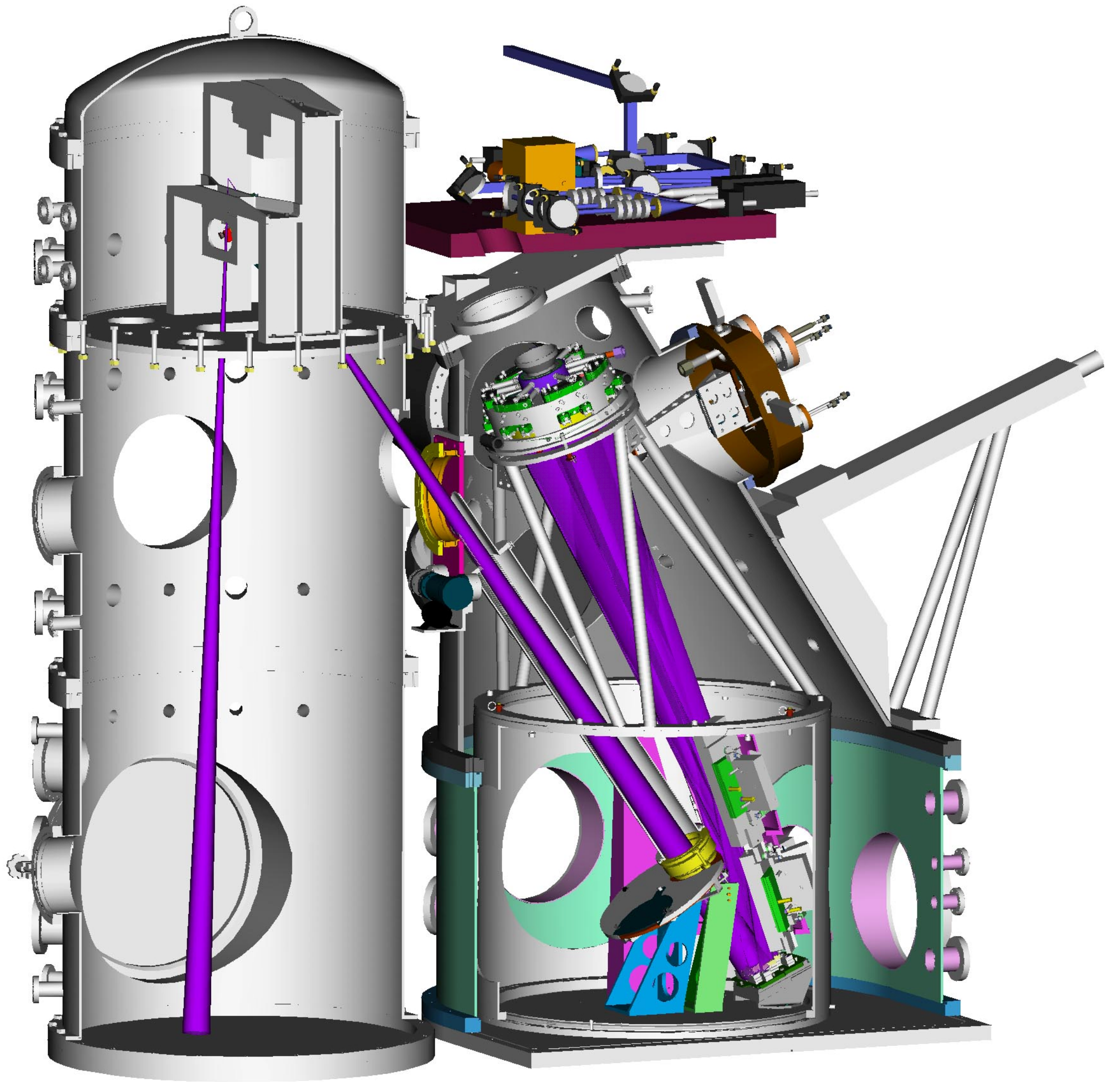
$$g(x) \geq 0$$ Inequality constraints

$$L = f(x) + y^T h(x) - w^T g(x)$$

Classes of Optimization Problems

- Unconstrained optimization
- Bound constrained optimization
  - Only upper and lower bounds
  - Sometimes called “box” constraints
- General nonlinearly constrained optimization
  - Equality and inequality constraints
  - Usually nonlinear
- Some special case classes
  - Linear programming (function and constraints linear)
  - Quadratic programming (quadratic function, linear constraints)

Parameter identification example

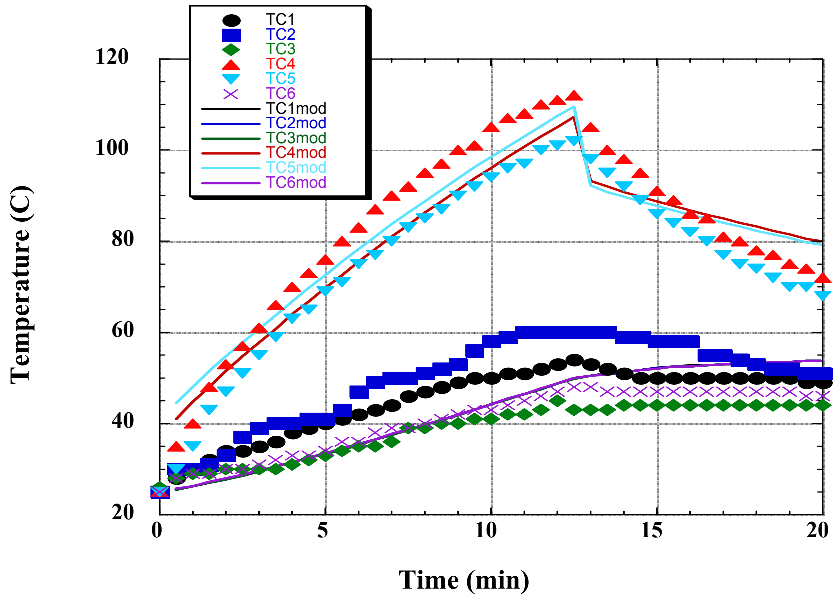


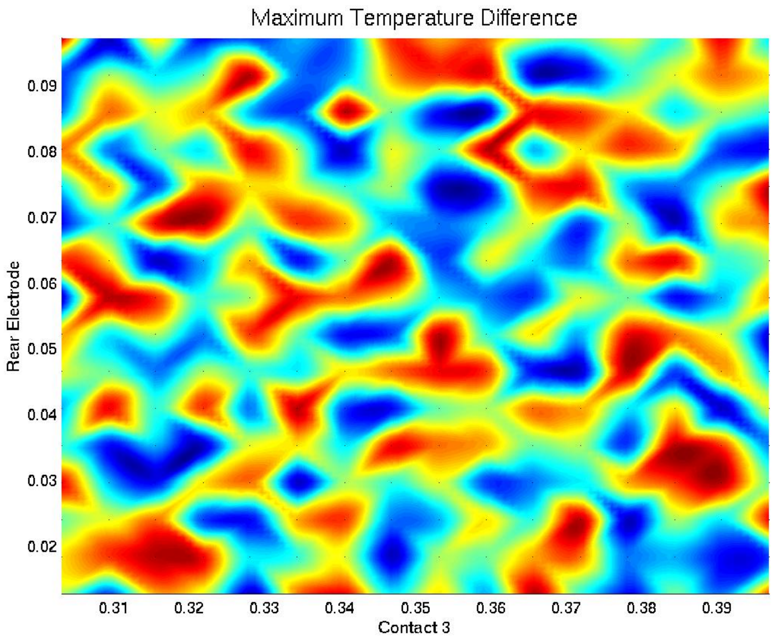
$$\min_{x} \quad \sum_{i=1}^{N} (T_i(x) - T_i^*)^2$$

$$\text{s.t.} \quad 0 \le x \le u$$

- Find model parameters, satisfying some bounds, for which the simulation matches the observed temperature profiles
- Computing objective function requires running thermal analysis code

Optimization formulation





- Objective function consists of computing the max temperature difference over 5 curves
- Each simulation requires approximately 7 hours on 1 processor
- Uncertainty in both the measurements and the model parameters

Some working assumptions

- Objective function is smooth
  - Usually true, but simulations can create noisy behavior
- Twice continuously differentiable
  - Usually true, but difficult to prove
- Constraints are linearly independent
  - Users can sometimes overspecify or incorrectly guess constraints
- Small dimensional, but expensive objective functions

OPT++ Philosophy

- Problem should be defined in terms the user understands
  - Do I have second derivatives available? vs. Is my objective function twice continuously differentiable?
- Solution methods should be easily interchangeable
  - Once the problem is setup, methods should be easy to interchange so that the user can compare algorithms
- Common components of methods should be interchangeable
  - Algorithm developers should be able to re-use common components from other algorithms

Problem Classes in OPT++

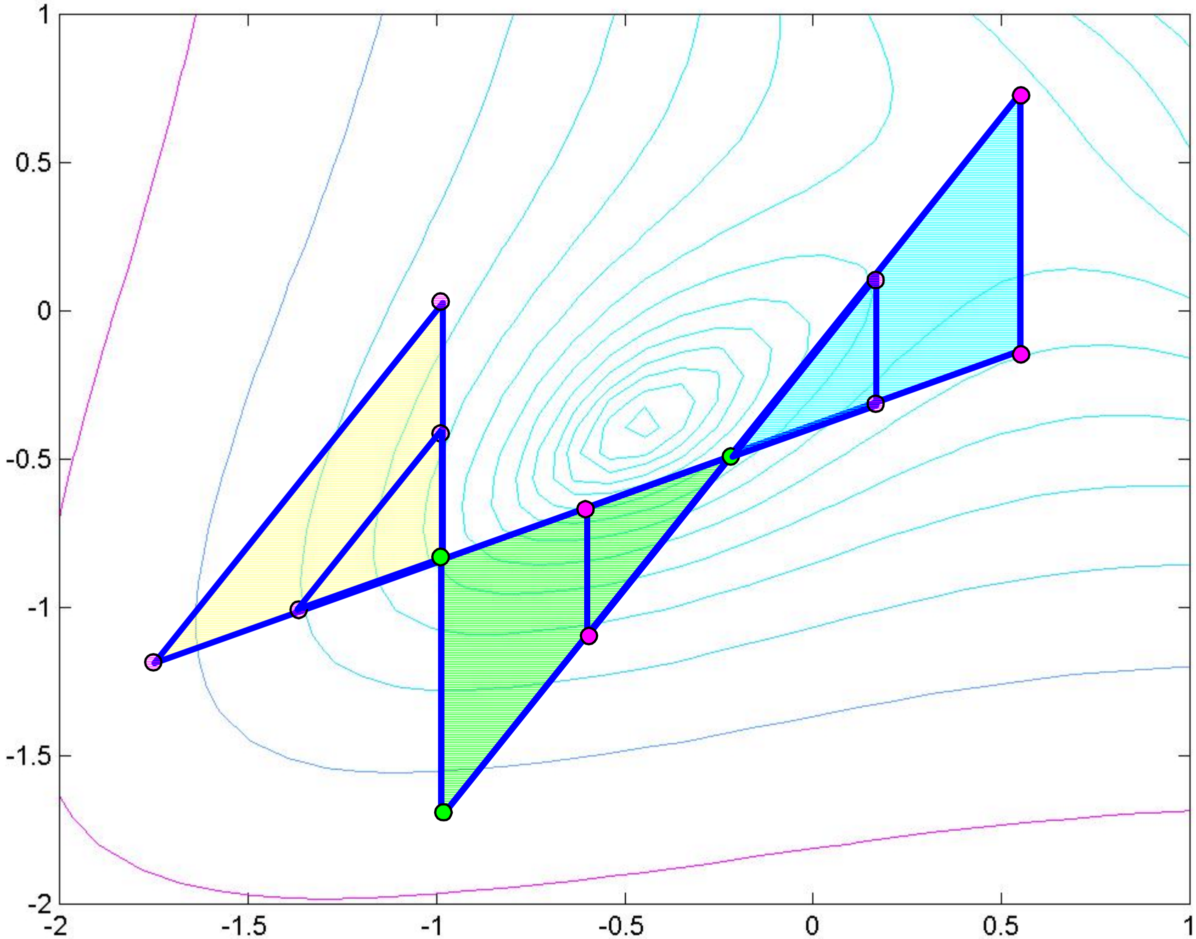
- Four major classes of problems available
  - *NLF0(ndim, fcn, init_fcn, constraint)*
    - Basic nonlinear function, no derivative information available
  - *NLF1(ndim, fcn, init_fcn, constraint)*
    - Nonlinear function, first derivative information available
  - *FDNLF1(ndim, fcn, init_fcn, constraint)*
    - Nonlinear function, first derivative information approximated
  - *NLF2(ndim, fcn, init_fcn, constraint)*
    - Nonlinear function, first and second derivative information available

Classes of Solvers in OPT++

- Pattern search
  - No derivative information required
- Conjugate Gradient
  - Derivative information may be available but doesn't use quadratic information
- Newton-type methods
  - Algorithm attempts to use/approximate quadratic information
  - Newton
  - Finite-Difference Newton
  - Quasi-Newton
  - NIPS

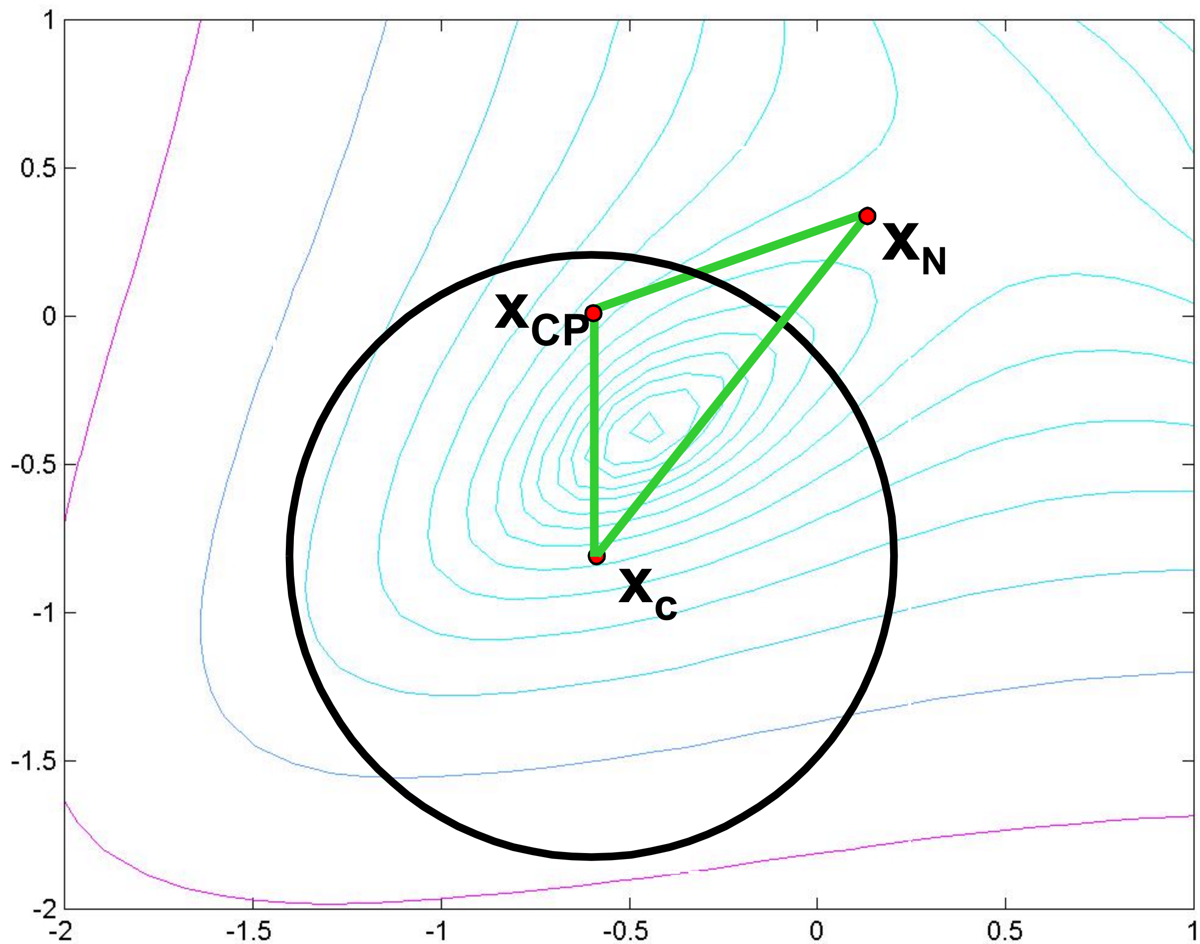
Quick tour of some of the algorithms

Pattern search



- Can handle noisy functions
- Do not require derivative information
- Inherently parallel
- Convergence can be painfully slow

Newton-type Methods



- Fast convergence properties
- Good global convergence properties
- Inherently serial
- Difficulties with noisy functions

NIPS: Nonlinear Interior Point Solver

- Interior point method
- Based on Newton's method for a particular system of equations (perturbed KKT equations, slack variable form)
- Can handle general nonlinear constraints
- Can handle strict feasibility

$$F(\mu) = \begin{bmatrix} \nabla f(x) + \nabla h(x) y - \nabla g(x) w \\ w - z \\ h(x) \\ g(x) - s \\ ZSe - \mu e \end{bmatrix} = 0$$

Constraints

- Constraint types
  - BoundConstraint(numconstraints, lower, upper)
  - LinearInequality(A, rhs, stdFlag)
  - NonLinearInequality(nlprob, rhs, numconstraints, stdFlag)
  - LinearEquation(A, rhs)
  - NonLinearEquation(nlprob, rhs, numconstraints)
- The whole shebang
  - CompoundConstraint(constraints)

Algorithm Choices Depend on Problem

	NLF0	FDNLF1	NLF1	NLF2
OptPDS	x	x	x	x
OptCG		x	x	x
OptQNewton		x	x	x
OptBCQNewton		x	x	x
OptFDNewton		x	x	x
OptFDNIPS		x	x	x
OptNewton				x
OptBCNewton				x
OptNIPS				x

Bare bones example: unconstrained optimization

```
void init_rosen(int ndim, ColumnVector& x);
void rosen(int ndim, const ColumnVector& x, double& fx, int& result);
int main() {
    int ndim = 2;
    FDNLF1 nlp(ndim, rosen, init_rosen);
    nlp.initFcn();
    OptQNewton objfcn(&nlp);
    objfcn.setSearchStrategy(TrustRegion);
    objfcn.setMaxFeval(200);
    objfcn.setFcnTol(1.e-4);
    objfcn.optimize();
}
```

Example 2: Constrained optimization

$$\min\ (x_1 - x_2)^2 + (1/9)(x_1 + x_2 - 10)^2 + (x_3 - 5)^2$$

$$s.t.$$

$$x_1^2 + x_2^2 + x_3^2 \le 48,$$

$$-4.5 \le x_1 \le 4.5,$$

$$-4.5 \le x_2 \le 4.5,$$

$$-5.0 \le x_3 \le 5.0$$

Constrained optimization (cont.)

```
int ndim = 3;

ColumnVector lower(ndim), upper(ndim);

lower << -4.5 << -4.5 << -5.0;    upper << 4.5 << 4.5 << 5.0 ;

Constraint c1 = new BoundConstraint(ndim, lower, upper);

// Nonlinear inequality constraint

NLP* chs65 = new NLP(new NLF2(ndim, 1, ineq, hs65,
    init_hs65));

Constraint nleqn = new NonLinearInequality(chs65);

// Put everything together in one constraint object

CompoundConstraint* constraints = new
        compoundConstraint(nleqn, c1);
```

Constrained optimization (cont.)

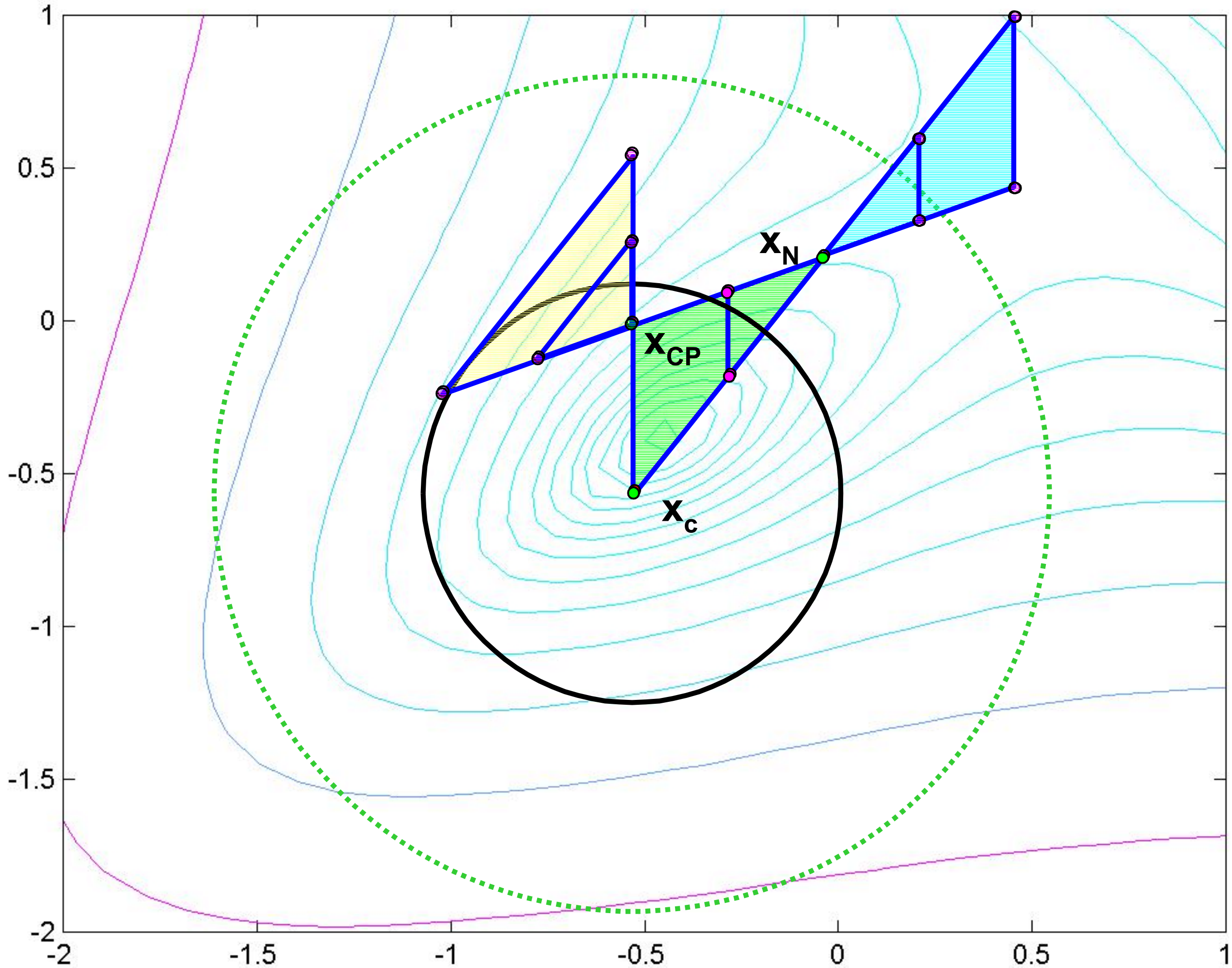
```
// Put it all together
NLF2 nips(ndim, hs65, init_hs65, constraints);
nips.initFcn();
// Define the optimization object
OptNIPS objfcn(&nips);
// Set tolerances and parameters
objfcn.setFcnTol(1.0e-06);
objfcn.setMaxIter(150);
objfcn.setMeritFcn(ArgaezTapia);
objfcn.optimize();
```

Parallel Optimization

Schnabel (1995) Identified Three Levels for Introducing Parallelism Into Optimization

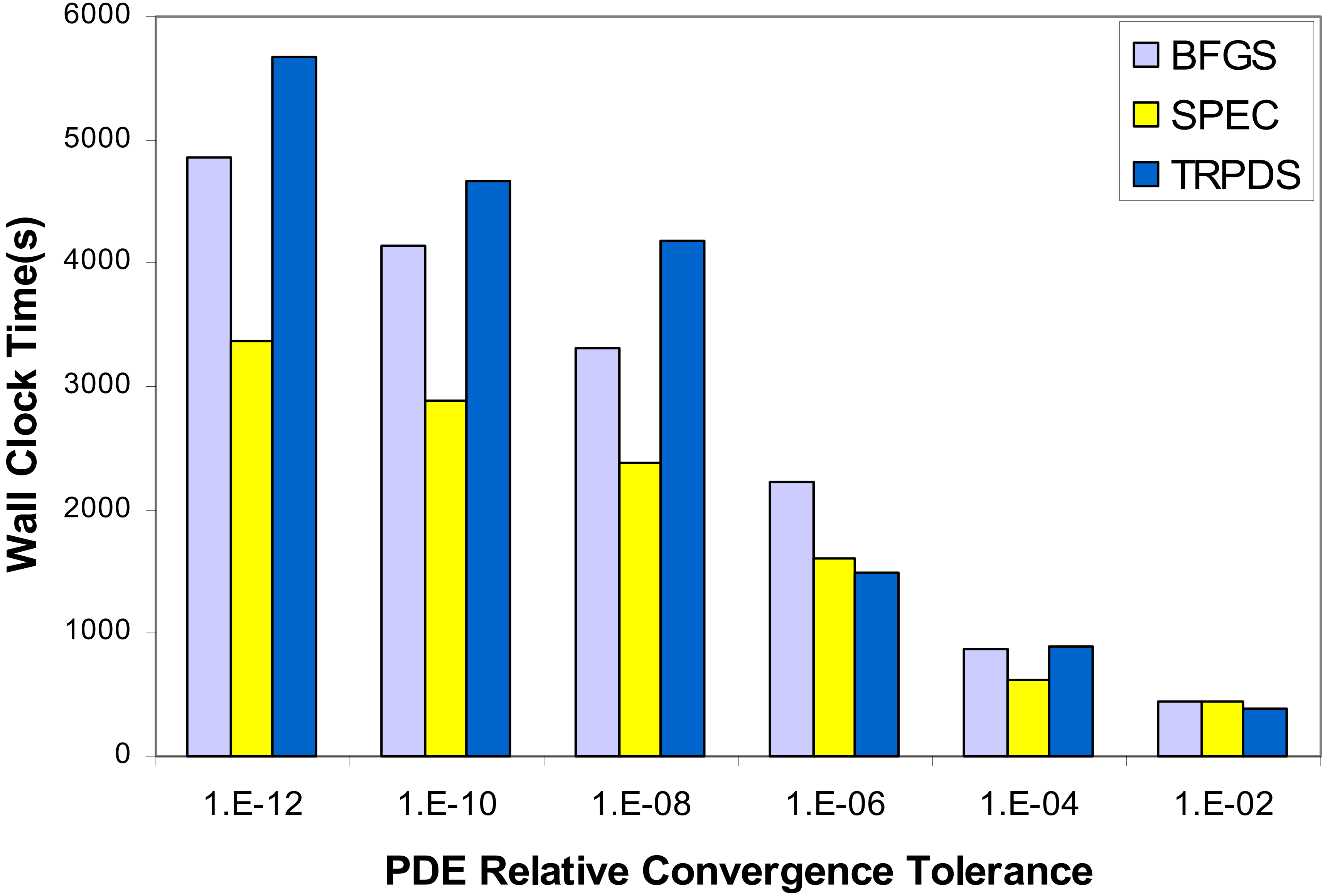
- Parallelize evaluation of function/gradient/constraints
  - May or may not be easy to implement
- Parallelize linear algebra
  - Really only useful if the optimization problem is large-scale
- Parallelize optimization algorithm at a high level
  - Multiple function evaluations in parallel

Trust Region with PDS



- Fast convergence properties of Newton method
- Good global convergence properties of trust region approach
- Inherent parallelism of PDS
- Ability to handle noisy functions

Comparison of TRPDS with other approaches



Summary

- OPT++ can handle many types of nonlinear optimization problems
- The toolkit can be used to compare the effectiveness of several algorithms on the same problem easily
- The user needs to provide only functions for the objective function and the constraints
  - If additional information is available it can be easily incorporated
- The code is open source and available at either
  - http://www.nersc.gov/~meza/projects/opt++
  - http://csmr.ca.sandia.gov/opt++

References

- Other links
  - http://sal.kachinatech.com/B/3/index.shtml
  - http://www-neos.mcs.anl.gov/neos
  - http://www.mcs.anl.gov/tao
  - http://endo.sandia.gov/DAKOTA/index.html
- Books/Papers
  - Dennis and Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, 1983
  - Gill, Murray, Wright, *Practical Optimization*, Academic Press, 1981
  - El-Bakry, Tapia, Tsuchiya, Zhang, *On the Formulation and Theory of the Newton Interior-Point Method for Nonlinear Programming*, JOTA, Vol. 89, No.3, pp.507-541, 1996
  - More´ and Wright, Optimization Software Guide, SIAM, 1993